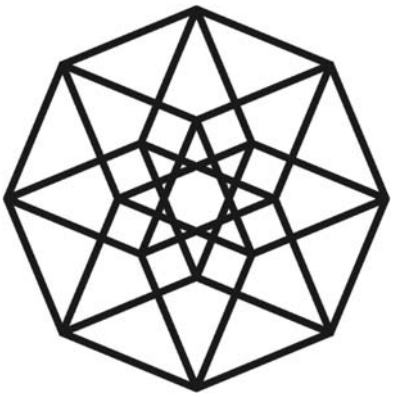
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Высшая математика»

ПРОГРАММИРОВАНИЕ

Методические рекомендации к лабораторным работам для студентов направления подготовки 01.03.04 «Прикладная математика» очной формы обучения

Часть 2



Могилев 2021

УДК 004.4
ББК 32.973
П78

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Высшая математика» «27» мая 2021 г., протокол № 9

Составители: ст. преподаватель А. Н. Бондарев;
доц. И. И. Маковецкий;
доц. Д. В. Роголев

Рецензент канд. техн. наук, доц. В. М. Ковальчук

Методические рекомендации разработаны на основе рабочей программы по дисциплине «Программирование» для студентов направления подготовки 01.03.04 «Прикладная математика» и предназначены для использования при проведении лабораторных работ во втором семестре.

Учебно-методическое издание

ПРОГРАММИРОВАНИЕ

Часть 2

Ответственный за выпуск	В. Г. Замураев
Корректор	Т. А. Рыжикова
Компьютерная вёрстка	Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 56 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

Часть 2

Введение

Целью изучения дисциплины «Программирование» является формирование у студентов базовых знаний в программировании, развитие навыков постановки, формализации и решения задачи на языке программирования высокого уровня.

Во втором семестре, в соответствии с рабочей программой дисциплины, студенты выполняют 10 лабораторных работ, варианты и условия которых приведены в методических рекомендациях.

К защите каждой лабораторной работы студент готовит отчёт, который оформляется с использованием текстовых редакторов и включает название и цель работы, формулировку задачи для своего варианта, алгоритм решения, текст программы и результаты ее тестирования.

Программы пишутся на языке Python. Оценивание осуществляется автоматически отправкой исполняемого кода в качестве ответа на тестовое задание.

При подготовке к защите студенту рекомендуется изучить теоретический материал и ответить на контрольные вопросы.

Лабораторная работа № 11. Работа с кортежами

Цель работы: изучить особенности типа данных tuple в Python, ознакомиться с основными методами кортежей.

11.1 Теоретическая часть

Кортежи – это неизменяемые последовательности, обычно используемые для хранения коллекций разнородных данных. Кортежи также используются в тех случаях, когда требуется неизменяемая последовательность однородных данных. В python кортежи представлены классом tuple().

Кортежи могут быть созданы несколькими способами:

– с помощью пары скобок для обозначения пустого кортежа `()`;

– с помощью запятой для одиночного кортежа `a,` или `(a,)`;

– разделением элементов запятыми: `a, b, c` или `(a, b, c)`;

– с помощью встроенного класса tuple(): `tuple()`, `tuple(iterable)`.

Конструктор класса tuple() создает кортеж, элементы которого совпадают и находятся в том же порядке, что и элементы итератора iterable. Аргумент iterable может быть либо последовательностью, контейнером, поддерживающим итерацию, либо объектом итератора.

Обратите внимание, что запятая создает кортеж. Скобки необязательны, за исключением случая пустого кортежа, или в случае, когда они необходимы, чтобы избежать синтаксической двусмысленности.

Кортежи относятся к неизменяемым типам, особенность машинного представления кортежей позволяет использовать для их хранения меньше оперативной памяти.

Операции над кортежами в порядке возрастания приоритета:

проверка существования значения в кортеже `x in tpl` – возвращает логическое значение True, если элемент x присутствует в кортеже tpl, False – в противном случае;

сложение двух кортежей `tpl1 + tpl2` – возвращает новый кортеж, содержащий последовательно элементы кортежей tpl1, tpl2;

повторение кортежа n раз `tpl * n` – возвращает новый кортеж, в котором кортеж tpl повторяется n раз;

получение элемента кортежа по индексу `tpl[i]` – возвращает элемент кортежа с индексом i либо ошибку, если индекс вышел за пределы кортежа;

получение среза кортежа `tpl[i:j:k]` – возвращает новый кортеж, содержащий элементы кортежа tpl начиная от индекса i до индекса j с шагом k;

вычисление длины кортежа `len(tpl)` – возвращает количество элементов в кортеже;

наименьшее значение элемента кортежа `min(tpl)` – возвращает минимальный элемент кортежа;

наибольшее значение элемента кортежа `max(tpl)` – возвращает максимальный элемент кортежа;

удаление кортежа `del tpl` – удаляет кортеж tpl из памяти;

метод index() – возвращает номер первого вхождения элемента в кортеж: `tpl.index(x);`

метод count() – подсчитывает количество вхождений элемента в кортеж: `tpl.count(x).`

Полезным свойством кортежей является распаковка, которая позволяет выделять содержимое кортежей в другие элементы. Пример:

```
tpl = (1, 2, 3,)
a, b, c = tpl
```

При выполнении этого фрагмента кода переменные получат значения a = 1, b = 2, c = 3.

С помощью распаковки операция перестановки значений переменных выполняется в одно действие:

```
a, b = b, a
```

Язык программирования Python обладает расширенными возможностями распаковки. В случае, если количество распаковываемых значений больше, чем переменных, перед одной из переменных распаковки ставится символ * (звездочка), это позволяет распаковывать последовательности, длина которых заранее неизвестна.

```
tpl = (1, 2, 3, 4, 5)
a, *b, c = tpl
```

Выполнение данного фрагмента кода присвоит переменным значения a = 1, b = [2, 3, 4], c = 5.

Для ввода кортежа с клавиатуры необходимо использовать функцию eval().

11.2 Контрольные вопросы

1 Что такое тип данных tuple?

2 Перечислите способы генерации кортежей.

3 Какой тип будет иметь переменная a, если `a = (1)`?

4 Какой метод типа tuple() позволяет определить количество вхождений некоторого элемента в кортеж?

5 Какой метод типа tuple() позволяет определить индекс вхождения некоторого элемента в кортеж?

6 Какая команда возвращает количество элементов кортежа?

7 Каким будет результат выполнения команды `a, b, c = (1, 2, 3, 4)`?

8 Каким способом можно удалить элемент кортежа?

9 Для чего служит команда `x in tpl`?

11.3 Задания для самостоятельного выполнения

1 Напишите программу на языке программирования Python, которая генерирует кортеж, состоящий из четных чисел от 2 до 20000.

2 Напишите программу на языке программирования Python, которая организует ввод чисел с клавиатуры и сохраняет их в кортеж, окончанием ввода является ввод числа –1.

3 Заданы два кортежа одинаковых размерностей. Напишите программу на языке программирования Python, которая создает кортеж, элементами которого являются кортежи – пары соответствующих элементов из двух заданных.

4 Алгебраические векторы в пространстве заданы с помощью трехэлементных кортежей. Напишите программу на языке программирования Python, которая реализует операции сложения и умножения вектора на число, возвращающие новые кортежи, соответствующие результатам выполнения операций.

5 Алгебраические векторы в пространстве заданы с помощью трехэлементных кортежей. Напишите программу на языке программирования Python, которая реализует операцию скалярного произведения двух векторов.

6 Алгебраические векторы в пространстве заданы с помощью трехэлементных кортежей. Напишите программу на языке программирования Python, которая реализует операцию векторного умножения двух векторов, возвращая кортеж, соответствующий вектору произведения.

7 Алгебраические векторы в пространстве заданы с помощью трехэлементных кортежей. Напишите программу на языке программирования Python, которая реализует смешанное произведение трех векторов.

8 Напишите программу на языке программирования Python, которая выводит True, если все элементы кортежа одинаковые, и False – в противном случае.

9 Задан кортеж, содержащий цифры от 0 до 9. Напишите программу на языке программирования Python, которая преобразует кортеж в натуральное число, цифрами которого являются числа из кортежа.

10 Напишите программу на языке программирования Python, которая организует ввод кортежа с произвольными элементами с клавиатуры.

11 С клавиатуры последовательно вводятся кортежи, содержащие числа, окончанием ввода является ввод пустой строки. Напишите программу на языке программирования Python, которая выводит на экран максимальную из сумм элементов кортежей.

12 С клавиатуры последовательно вводятся кортежи (n-мерные векторы), содержащие числа, окончанием ввода является ввод пустой строки. Напишите программу на языке программирования Python, которая выводит на экран кортеж-вектор, длина которого наибольшая (вычисляется как евклидова норма) среди введенных кортежей-векторов.

13 С клавиатуры вводится кортеж, содержащий некоторые числа. Напишите программу на языке программирования Python, которая выводит кортеж, содержащий накопленные суммы элементов кортежа, введенного с клавиатуры.

14 Напишите программу на языке программирования Python, которая удаляет из заданного кортежа элемент с индексом n и возвращает полученный кортеж. Предусмотрите обработку возможных ошибок.

15 С клавиатуры вводится некоторое целое число (угол в градусах) и кортеж, содержащий два числа (координаты точки на плоскости). Напишите

программу на языке программирования Python, которая выводит кортеж – координаты точки, полученной из заданной поворотом на заданный угол.

16 Напишите программу на языке программирования Python, которая для введенного с клавиатуры кортежа из трех чисел (длины сторон треугольника) возвращает его площадь или сообщение о том, что треугольник не существует.

17 С клавиатуры вводятся кортежи, содержащие некоторые числа. Напишите программу на языке программирования Python, которая генерирует кортеж, содержащий в качестве элементов суммы кортежей, введенных с клавиатуры.

18 С клавиатуры вводится кортеж, содержащий положительные целые числа. Напишите программу на языке программирования Python, которая преобразует кортеж в целое число: (1, 23, 456) в 123456.

19 Задан кортеж, содержащий числовые кортежи. Напишите программу на языке программирования Python, которая возвращает кортеж, содержащий средние значения элементов кортежа.

20 Задан числовой кортеж с положительными элементами. Напишите программу на языке программирования Python, которая вычисляет среднее геометрическое его элементов.

Лабораторная работа № 12. Работа со списками

Цель работы: изучить особенности типа данных list в Python, ознакомиться с основными методами списков.

12.1 Теоретическая часть

Списки в Python – упорядоченные изменяемые коллекции объектов произвольных типов (в отличие от массивов). В Python списки представлены типом данных list().

Списки могут быть созданы несколькими способами:

– преобразованием любого итерируемого объекта в тип list();

– записью с помощью литерала в прямоугольных скобках;

– с помощью генераторов list comprehension:

```
new_list = [expr for member in iterable (if condition)].
```

Для обращения к элементам списка используются индексы, которые представляют номер элемента списка. Индексация элементов списка начинается с нуля. Допускается использовать отрицательную индексацию для доступа к элементам, расположенным в конце списка.

Списки являются итерируемыми объектами, перебор которых может быть организован в циклах while или for.

Два списка считаются равными, если они содержат один и тот же набор элементов.

Методы списков:

append(item) – добавляет элемент item в конец списка;

insert(index, item) – добавляет элемент item в список по индексу index;

remove(item) – удаляет элемент item, причем удаляется только первое вхождение элемента, если такой элемент отсутствует в списке, генерируется исключение;

clear() – удаляет все элементы списка;

index(item) – возвращает индекс элемента item, если элемент не найден, генерирует исключение;

pop([index]) – удаляет и возвращает элементы по индексу index либо удаляет последний элемент, если индекс не задан;

count(item) – возвращает количество вхождений элемента item в список;

sort([key], [reverse]) – сортирует элементы по возрастанию, по убыванию или по заданной функции key, изменяя список;

reverse() – расставляет элементы списка в обратном порядке, изменяя сам список.

Функции для работы со списками:

len(list) – возвращает количество элементов списка list;

sorted(list, [key]) – возвращает сортированный список, не изменяя самого списка;

min(list) – возвращает наименьший элемент списка;

max(list) – возвращает наибольший элемент списка.

Для проверки наличия элемента в списке используется оператор принадлежности in.

При работе со списками следует учитывать, что списки представляют собой изменяемый тип, поэтому при копировании двух списков присваиванием (мелкое копирование) обе переменные будут указывать на один и тот же список, поэтому изменение одной переменной затронет и другую. Для того чтобы осуществить глубокое копирование списков, следует использовать метод deepcopy() модуля copy.

Мелкое копирование можно выполнить с помощью присваивания новой переменной среза или с использованием метода класса list.copy().

12.2 Контрольные вопросы

1 Что такое списки в языке Python?

2 Какими способами можно задать списки в Python?

3 Каким образом можно перебрать все элементы списка?

4 Объясните результат выполнения команды a=[2]*7.

5 Каким будет значение переменной d после выполнения команды d=list(range(10))?

6 Каким будет значение переменной d после выполнения команды d=list(range(10,1,-1))?

7 Чем отличается поверхностное копирование списков от глубокого?

12.3 Задания для самостоятельного выполнения

1 Напишите программу циклического сдвига элементов списка влево. Например, дан список [1, 2, 3, 4, 5, 6], в результате выполнения программа возвращает список [2, 3, 4, 5, 6, 1].

2 Напишите программу, которая из списка телефонных номеров [‘+375231456’, ‘+7915235612’,’+374235626’,’+3756267221’,’+42626473721’] удаляет все номера с кодом «+375».

3 Пользователь вводит с клавиатуры последовательность чисел до тех пор, пока не введет число 0. Напишите программу на языке программирования Python, которая формирует список, состоящий из квадратов введенных чисел.

4 С клавиатуры вводится список чисел (использовать функцию eval()), необходимо проверить, содержит ли список число 5.

5 Напишите программу скалярного умножения векторов, заданных списками координат.

6 Пользователь вводит с клавиатуры N чисел (задается в программе). Напишите программу на языке программирования Python, которая формирует список, состоящий из продублированных элементов. Например, пользователь вводит 1, 2.8, ‘abc’, в результате выполнения программа формирует список [1, 1, 2.8, 2.8, ‘abc’, ‘abc’].

7 Дан список [–1, 0, 5, 4, 2]. Напишите программу, которая увеличивает каждый элемент на 7.2.

8 Дан список, содержащий более одного числа. Напишите программу на языке программирования Python, которая генерирует новый список, состоящий из разностей соседних элементов заданного списка. Использовать list comprehension.

9 Список содержит произвольные числа. Напишите программу на языке программирования Python, которая формирует новый список из элементов заданного, которые больше по модулю заданного числа M. Вывести на экран число M, данный и полученный списки.

10 Задан список, содержащий целые числа. Напишите программу на языке программирования Python, которая все отрицательные элементы заменяет их модулями, а положительные элементы возводит в квадрат. Вывести на экран исходный список и полученный.

11 Написать программу на языке программирования Python, которая определяет, присутствуют ли в списке одинаковые элементы, и если да, то выводит их на экран.

12 Напишите программу на языке программирования Python, которая в заданном списке, содержащем числа, меняет местами наибольший и наименьший элементы.

13 С клавиатуры вводится строка, представляющая собой арифметическое выражение с целыми числами (содержит операции сложения, вычитания, умножения и деления, не содержит скобок). Напишите программу на языке программирования Python, которая преобразует введенную строку в список, каждый элемент которого является операндом или оператором введенного

арифметического выражения. Например: ‘12+7-2*9’ соответствует списку [12, ’+’, 7, ’–‘, 2, ’*’, 9].

14 Напишите программу на языке программирования Python, которая преобразует заданный список строк таким образом, чтобы все слова в списке имели одинаковую длину, равную длине самого длинного элемента списка. Короткие слова дополняются символами подчеркивания справа до получения требуемой длины.

15 Напишите программу на языке программирования Python, которая сортирует список, состоящий из чисел, по возрастанию их абсолютной величины.

16 Напишите программу на языке программирования Python, которая сортирует список, состоящий из строк, по убыванию их длины.

17 Два числовых множества заданы списками, содержащими их элементы. Напишите программу на языке программирования Python, которая возвращает список, содержащий элементы пересечения заданных множеств, отсортированный по возрастанию.

18 Два числовых множества заданы списками, содержащими их элементы. Напишите программу на языке программирования Python, которая возвращает список, содержащий элементы разности заданных множеств, отсортированный по возрастанию.

19 Список содержит кортежи, состоящие из трех чисел. Напишите программу на языке программирования Python, которая сортирует список по вторым элементам кортежей.

20 В некоторой области n городов, расположенных вдоль одной дороги. Гражданская оборона области построила вдоль этой же дороги m бомбоубежищ. Известны расстояния городов от начала дороги (набор из n чисел) и расстояния бомбоубежищ от начала дороги (набор из m чисел). Напишите программу на языке программирования Python, которая выводит для каждого города номер в списке ближайшего к нему бомбоубежища (порядковый номер в наборе из m чисел).

21 С клавиатуры вводится невозрастающая последовательность натуральных чисел – рост учеников класса при построении на уроке физкультуры. После этого вводится число X – рост Пети. Напишите программу на языке программирования Python, которая определяет, каким по счету должен стоять Петя в строю. Если в строю есть люди с одинаковым ростом, таким же, как и у Пети, то он должен встать после них.

22 Известно, что на шахматной доске 8x8 можно расставить восемь ферзей так, чтобы они не били друг друга. С клавиатуры вводится восемь пар чисел, каждое от 1 до 8 – координаты восьми ферзей на шахматной доске. Напишите программу на языке программирования Python, которая выводит NO, если ферзи не бьют друг друга, и YES – иначе.

Лабораторная работа № 13. Работа со словарями

Цель работы: изучить тип данных dictionary, его методы и функции для работы с ним.

13.1 Теоретическая часть

Словарь в Python – тип данных, в котором хранится коллекция элементов. Каждый элемент в словаре имеет уникальный ключ, с которым ассоциировано некоторое значение.

Определение словаря имеет следующий синтаксис:

```
dict = { key1: value1, key2: value2, … }
```

В словаре ключи необязательно должны принадлежать к одному типу данных, единственное ограничение на ключи – они должны быть хэшируемыми. В словаре не может быть двух одинаковых ключей. Словарь может быть пустым.

Также можно получить словарь преобразованием списка или кортежа, содержащих пары элементов преобразованием типа dict().

Доступ к элементам словаря осуществляется по ключу dict[key]. Операция dict[key] = value выполняется в случае, если ключ key отсутствует в словаре, происходит добавление элемента в словарь.

В случае, если происходит обращение по ключу, отсутствующему в словаре, Python сгенерирует ошибку. Во избежание подобной ситуации следует использовать метод словарей dict.get(key, default), который возвращает элемент словаря с индексом key или значение default, если такой элемент отсутствует.

Удаление элемента словаря с заданным ключом выполняется с помощью оператора del dict[key].

Методы словарей:

dict.pop(key, default) – осуществляется удаление элемента с индексом key и его возврат. При отсутствии такого элемента возвращается значение default;

dict.clear() – осуществляет удаление словаря целиком;

dict.copy() – осуществляет копирование словаря. Данный метод осуществляет поверхностное копирование словаря. Для глубокого копирования следует использовать метод deepcopy() модуля copy;

dict.update(other_dict) – осуществляет добавления словаря other_dict к словарю dict, при этом изменяется только словарь dict;

dict.keys() – возвращает перечисляемый набор ключей словаря;

dicst.values() – возвращает перечисляемый набор значений словаря.

Допускается хранить в словарях элементы любых типов, в том числе и словари, для доступа к значениям таких словарей необходимо указывать соответствующее количество ключей.

13.2 Контрольные вопросы

1 Что представляет собой тип «словарь» в языке программирования Python?

2 Какими способами можно получить доступ к значению ключа?

3 Что может служить ключом словаря?
4 Для чего используется метод pop()?
5 Что возвращают методы keys(), items(), values()?
6 Каким способом можно копировать словарь?

13.3 Задания для самостоятельного выполнения

1 Напишите программу на языке программирования Python, которая по заданному словарю «ключ: значение» строит словарь «значение: ключ». Предполагается, что все значения в словаре различные.

2 Дан текст на некотором языке. Напишите программу на языке программирования Python, которая подсчитывает, сколько раз каждое слово входит в этот текст, и выводит десять самых часто употребляемых слов в этом тексте и количество их употреблений.

3 Дан список стран и языков, на котором говорят в этой стране в формате <название страны>: <язык1> <язык2>… На ввод задается N – длина списка и список языков. Напишите программу на языке программирования Python, которая для каждого языка из введенного списка выводит, в каких странах на нем говорят.

4 С клавиатуры вводится число N – длина словаря – и вводится словарь, содержащий N пар слов. Каждое слово является синонимом к парному ему слову. Все слова в словаре различны. Напишите программу на языке программирования Python, которая для слова, полученного с клавиатуры, определяет его синоним.

5 Как известно, в США президент выбирается не прямым голосованием, а путем двухуровневого голосования. Сначала проводятся выборы в каждом штате и определяется победитель выборов в данном штате. Затем проводятся государственные выборы: на этих выборах каждый штат имеет определенное число голосов – число выборщиков от этого штата. На практике все выборщики от штата голосуют в соответствии с результатами голосования внутри штата, т. е. на заключительной стадии выборов в голосовании участвуют штаты, имеющие различное число голосов.

В первой строке дано количество записей. Далее, каждая запись содержит фамилию кандидата и число голосов, отданных за него в одном из штатов. Подведите итоги выборов: для каждого из участника голосования определите число отданных за него голосов. Участников нужно выводить в алфавитном порядке.

Тест:
5
McCain 10
McCain 5
Obama 9
Obama 8
McCain 1

Правильный ответ:
McCain 16
Obama 17

6 В файловую систему одного суперкомпьютера проник вирус, который сломал контроль за правами доступа к файлам. Для каждого файла известно, с какими действиями можно к нему обращаться: write – W, read – R, execute – X. С клавиатуры вводится число N – количество файлов, содержащихся в данной файловой системе. Далее вводится N имен файлов и допустимых с ними операций, разделенные пробелами. Далее вводится число M – количество запросов к файлам. Вводится M строк – запросы вида Операция Файл. К одному и тому же файлу может быть применено любое количество запросов. Напишите программу на языке программирования Python, которая восстанавливает права для доступа к файлам – возвращает OK, если над файлом выполняется допустимая операция или Access Denied, если операция недопустима.

Тест:
2
abacaba X R
nonono W
3
read abacaba
write nonono
execute nonono
Правильный ответ:
OK
OK
Access Denied

7 Дана база данных о продажах некоторого интернет-магазина. С клавиатуры вводится число N – количество записей в базе данных и затем N строк вида «Покупатель товар количество», где Покупатель – имя покупателя (строка без пробелов), товар – наименование товара (строка без пробелов), количество – количество приобретенных единиц товара.

Напишите программу на языке программирования Python, которая выводит список всех покупателей, а также для каждого покупателя количество приобретенных им единиц каждого вида товаров. Список покупателей и список товаров для покупателя должен выводиться в лексикографическом порядке.

Тест:
3
Ivanov paper 3
Ivanov paper 5
Ivanov marker 3
Правильный ответ:
Ivanov:
marker 3
paper 8

8 В генеалогическом древе у каждого человека, кроме родоначальника, есть ровно один родитель. Каждому элементу дерева сопоставляется целое неотрицательное число, называемое высотой. У родоначальника высота равна 0, у любого другого элемента высота на 1 больше, чем у родителя.

С клавиатуры вводится натуральное число N – количество элементов в генеалогическом древе. Далее вводится N-1 строка, задающая родителя для каждого элемента древа, кроме родоначальника. Каждая строка имеет вид имя_потомка имя_родителя.

Напишите программу на языке программирования Python, которая выводит список всех элементов древа в лексикографическом порядке и для каждого имени его высоту.

Тест:

9
Мария Михаил_Федорович
Алексей Михаил_Федорович
Наталья Михаил_Федорович
Федор Алексей
Иоанн Алексей
Петр Алексей
Екатерина Алексей
Анна Иоанн

Правильный ответ:

Алексей 1
Анна 3
Екатерина 2
Иоанн 2
Мария 1
Михаил_Федорович 0
Наталья 1
Петр 2
Федор 2

Лабораторная работа № 14. Работа с двумерными массивами

Цель работы: научиться использовать встроенные типы данных языка программирования Python для организации хранения и работы с двумерными массивами.

14.1 Теоретическая часть

В языке программирования Python не предусмотрено стандартных функций для организации многомерных массивов, поэтому для организации двумерных массивов необходимо использовать тип данных list(), список внутри списка.

Для доступа к элементам двумерного массива необходимо использовать

двойную индексацию, вложенные списки позволяют использование методов и функций списков. Например, команда

```
a = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

создаст массив размерности 3x3, заполненный последовательными числами от 1 до 9 по строкам. Для того чтобы обратиться к элементу, стоящему на пересечении 1 строки и 3 столбца, необходимо использовать команду `a[0][2]`, поскольку индексация элементов списков начинается с нуля.

Двумерные массивы поддерживают те же функции и методы, что и списки.

14.2 Контрольные вопросы

1 Какой тип данных Python необходимо использовать для организации данных в формате массива?

2 Можно ли организовать с помощью средств языка программирования Python трехмерные массивы?

3 Какой метод типа данных list() необходимо использовать для того, чтобы удалить строку массива?

4 Если a – двумерный массив размерностью 3x4, то каким будет результат выполнения команды `print(a[3][4])`?

5 Можно ли создать двумерный массив, в каждой строке которого различное число элементов?

14.3 Задания для самостоятельного выполнения

1 Напишите программу на языке программирования Python, которая позволяет ввести с клавиатуры двумерный числовой массив и вывести его в виде таблицы, в которой элементы каждого столбца расположены друг под другом. Программа получает на вход размеры массива n и m, затем n строк по m чисел в каждой.

2 С клавиатуры вводится натуральное число N. Напишите программу на языке программирования Python, которая строит двумерный массив, содержащий N^2 элементов, заполненный натуральными числами от 1 до N^2, расположенными по строкам.

3 С клавиатуры вводится размерность двумерного массива, а затем целочисленный массив. Напишите программу на языке программирования Python, которая подсчитывает количество элементов массива, больших, чем среднее значение элементов массива.

4 Для произвольного двумерного массива напишите программу на языке программирования Python, которая вычисляет сумму всех нечетных элементов.

5 Напишите программу на языке программирования Python, которая для произвольной числовой матрицы определяет номер столбца, сумма элементов которого наименьшая.

6 Напишите программу на языке программирования Python, которая генерирует и выводит на экран двумерный массив

01 02 03 04
12 13 14 05
11 16 15 06
10 09 08 07

7 Задан двумерный массив заданного размера. Напишите программу на языке программирования Python, которая строит новый массив, полученный из исходного делением на наибольший по модулю элемент.

8 Дан двумерный массив размерности mx2, в котором хранятся координаты точек на плоскости. Напишите программу на языке программирования Python, которая выводит координаты центра окружности, на которой лежит наибольшее количество точек из заданного массива.

9 Дана квадратная матрица размерности NxN. Напишите программу на языке программирования Python, которая зеркально отражает элементы матрицы относительно побочной диагонали.

10 Дана двумерная матрица размера MxN. Две строки считаются похожими, если в них содержатся одинаковые множества чисел. Напишите программу на языке программирования Python, которая выводит количество строк, которые похожи на последнюю.

11 Найдите индексы первого вхождения максимального элемента. Выведите два числа: номер строки и номер столбца, в которых стоит наибольший элемент в двумерном массиве. Если таких элементов несколько, то выводится тот, у которого меньше номер строки, а если номера строк равны, то тот, у которого меньше номер столбца. Программа получает на вход размеры массива n и m, затем n строк по m чисел в каждой.

12 Дано нечетное число n. Создайте двумерный массив из n×n элементов, заполнив его символами "." (каждый элемент массива является строкой из одного символа). Затем заполните символами "*" среднюю строку массива, средний столбец массива, главную диагональ и побочную диагональ. В результате единицы в массиве должны образовывать изображение звездочки. Выведите полученный массив на экран, разделяя элементы массива пробелами.

13 Даны два числа n и m. Создайте двумерный массив размером n×m и заполните его символами "." и "*" в шахматном порядке. В левом верхнем углу должна стоять точка.

14 Дано число n. Создайте массив размером n×n и заполните его по следующему правилу. На главной диагонали должны быть записаны числа 0, на двух диагоналях, прилегающих к главной, – числа 1, на следующих двух диагоналях – числа 2 и т. д.

15 Дано число n. Создайте массив размером n×n и заполните его по следующему правилу:

– числа на диагонали, идущей из правого верхнего в левый нижний угол, равны 1;

– числа, стоящие выше этой диагонали, равны 0;

– числа, стоящие ниже этой диагонали, равны 2;

Полученный массив выведите на экран. Числа в строке разделяйте одним пробелом.

16 Дан двумерный массив и два числа: i и j. Поменяйте в массиве столбцы с номерами i и j и выведите результат. Программа получает на вход размеры массива n и m, затем элементы массива, затем числа i и j.

Решение оформите в виде функции swap_columns(a, i, j).

Лабораторная работа № 15. Работа с датой и временем

Цель работы: изучить особенности работы с библиотеками time и datetime.

15.1 Теоретическая часть

В языке программирования Python нет встроенного типа данных для хранения даты и времени. Для работы с датой и временем в стандартной установке Python имеются стандартные модули time и datetime.

Модуль time предназначен для работы со временем. Класс time.struct_time – тип последовательности значения времени, который имеет интерфейс именованного кортежа, к элементам которого можно обращаться по индексу или имени:

0 или tm_year – год;

1 или tm_mon – номер месяца;

2 или tm_mday – номер дня в месяце;

3 или tm_hour – час;

4 или tm_min – минута;

5 или tm_sec – секунда;

6 или tm_wday – номер дня недели;

7 или tm_yday – номер дня в году;

8 или tm_isdst – параметр, определяющий необходимость перевода на летнее/зимнее время.

Основные классы и функции модуля **time**:

time.altzone – смещение часового пояса в секундах к западу от нулевого меридиана;

time.asctime(t) – преобразовывает тип struct_time в строку вида “Thu Sep 27 16:42:37 2021”, если аргумент не указан, использует текущее время;

time.ctime(t) – преобразует время в секундах с начала эпохи в строку вида “Thu Sep 27 16:42:37 2021”;

time.daylight – не 0, если определено, зимнее время или летнее;

time.gmtime(t) – преобразует время в секундах с начала эпохи в struct_time с параметром isdst=0;

time.localtime(t) – преобразует время в секундах с начала эпохи в struct_time с учетом локальных особенностей перевода времени;

time.mktime(t) – преобразует struct_time в число с начала секунд с начала эпохи;

time.sleep(t) – приостанавливает выполнение программы на t секунд;

time. strftime(format, t) – преобразует struct_time в строку по формату:

%a – сокращенное название дня недели;
%A – полное название дня недели;
%b – сокращенное название месяца;
%B – полное название месяца;
%c – дата и время;
%d – день месяца (от 0 до 31);
%H – час в 24-часовом формате (от 0 до 23);
%I – час в 12-часовом формате (от 1 до 12);
%j – день года (от 1 до 366);
%m – номер месяца (от 1 до 12);
%M – число минут (от 0 до 59);
%p – отметка ‘AM’ или ‘PM’;
%S – число секунд (от 0 до 59);
%U – номер недели в году (от 0 до 53);
%w – номер дня недели (от 0 до 6, 0 соответствует воскресенью);
%W – номер недели в году (от 0 до 53);
%x – дата;
%X – время;
%y – год без века (от 0 до 99);
%Y – год с веком;
%Z – временная зона;
%% – знак процента;

time.strptime(s, format) – разбор строки s, представляющей собой время в соответствии с форматом, возвращает значение struct_time;

time.time() – время, выраженное в секундах с начала эпохи;

time.timezone – смещение часового пояса в секундах относительно нулевого меридиана;

time.tzname – кортеж из двух строк, названия местного часового пояса.

Модуль **datetime** реализует следующие классы-типы:

datetime.date(year, month, day) – стандартная дата;

datetime.time(hour=0, minute=0, second=0, microsecond=0, tzinfo=None) – стандартное время, не зависит от даты;

datetime.datetime(year, month, day, hour=0, minute=0, second=0, microsecond=0, tzinfo=None) – класс – комбинация даты и времени;

datetime.timedelta() – разница между моментами времени с точностью до миллисекунд, допускает сложение и вычитание с типом datetime.datetime.

Методы класса datetime.datetime:

datetime.today() – возвращает объект datetime из текущей даты и времени;

datetime.fromtimestamp(t) – возвращает объект datetime из стандартного представления времени с учетом локальных особенностей;

datetime.fromordinal(t) – возвращает объект datetime из целого числа;

datetime.now() – возвращает объект datetime из текущей даты и времени с учетом локальных особенностей;

datetime.combine(date, time) – возвращает объект datetime из объектов

типов date и time;

datetime.strptime(date_string, format) – преобразует строку date_string в объект datetime по формату (аналогично модулю time);

datetime.strftime(format) – аналогично time.strftime();

datetime.replace(param=value) – возвращает новый объект datetime с измененными атрибутами;

datetime.timetuple() – возвращает объект struct_time из объекта datetime;

datetime.toordinal() – возвращает количество дней, прошедшее с 01.01.1970;

datetime.timestamp() – возвращает время в секундах с начала эпохи;

datetime.weekday() – день недели в виде числа, понедельник – 0, воскресенье – 6;

datetime.isoweekday() – день недели в виде числа, понедельник – 1, воскресенье – 7;

datetime.ctime() – аналогично time.ctime().

15.2 Контрольные вопросы

1 Для чего служит класс time.struct_time?

2 Назовите нижнюю границу изменения дат для модулей time и datetime.

3 Можно ли складывать две переменные типа datetime?

4 Для чего служит класс datetime.timedelta()?

15.3 Задания для самостоятельного выполнения

1 Напишите программу на языке программирования Python, которая реализует ввод с клавиатуры даты в формате «ДД.ММ.ГГГГ», сохранение в переменную типа datetime.datetime и вывод на экран в формате «ГГГГ-ММ-ДД».

2 С клавиатуры вводится последовательно N дат (N вводится с клавиатуры). Напишите программу на языке программирования Python, которая выводит две даты, между которыми наименьшее количество дней.

3 С клавиатуры вводится две даты. Напишите программу на языке программирования Python, которая выводит количество понедельников между этими датами.

4 Напишите программу, которая выводит на экран календарь. С клавиатуры вводятся год и месяц, для которых необходимо построить календарь.

5 С клавиатуры вводится дата; известно, что в этот день в университете была верхняя неделя. Напишите программу, которая выводит четыре следующих за текущей верхних недели (выводятся даты начала и конца недели).

6 С клавиатуры вводится дата рождения. Напишите программу на языке программирования Python, которая вычисляет количество прожитых секунд, минут, часов.

7 С клавиатуры вводится дата рождения. Напишите программу на языке программирования Python, которая выводит дату дня рождения, день недели которого совпадает с днем недели на момент рождения человека.

8 Напишите программу, которая по введенному с клавиатуры номеру дня с начала года определяет, на какой день недели приходится день с заданным номером. Предусмотрите вывод вариантов для високосного и невисокосного года.

9 С клавиатуры вводится дата. Напишите программу, которая выводит дату, следующую за введенной в том же формате.

10 С клавиатуры вводится дата в формате dd.mm. Напишите программу на языке программирования Python, которая определяет, сколько дней осталось до дня рождения на данный момент. Если сегодня – день рождения, то программа должна вывести 0.

11 С клавиатуры вводится две отметки времени в формате hh:mm:ss. Напишите программу на языке программирования Python, которая выводит, сколько времени (в формате hh:mm:ss) прошло от одной отметки до другой.

12 В часах села батарейка, и они стали идти вдвое медленней. Когда на часах было время h1:m1:s1, точное время было h2:m2:s2. Напишите программу на языке программирования Python, которая определяет точное время, когда часы в следующий раз покажут время h3:m3:s3.

Лабораторная работа № 16. Создание пользовательских функций

Цель работы: научиться использовать функциональное программирование в Python.

16.1 Теоретическая часть

Для определения функций пользователя существует два возможных подхода. В первом случае функция описывается с помощью набора инструкций def, во втором применяется так называемая lambda-функция.

Синтаксис набора инструкций def

```
def func_name(params, *add_params, **also_params):
    state1
    …
    state2
    return returned_value(s)
```

Синтаксис lambda-функции

```
lambda arguments: expression
```

Особенностью lambda-функций является то, что они определяются в одну строку, могут принимать любое количество аргументов (в том числе и ни одного). Например, функция возведения числа в квадрат может быть описана инструкцией

```
f = lambda x: x * x
```

Когда необходимо использовать функции, определяемые пользователем? В канонах программирования на Python имеется положение, которое утверждает,

что любой код, который приходится повторять в программе, следует оформить в виде функции, впоследствии вызывать ее в требуемом месте.

В язык программирования Python встроены несколько функций высших порядков (функции, принимающие в качестве аргументов другие функции): map(), filter(), reduce(), apply(). Функция map() предназначена для пакетного запуска функций, с ее помощью можно передать на вход пользовательской функции набор (кортеж) аргументов и получить набор (кортеж) результатов выполнения. Ознакомиться с действием остальных функций можно, используя систему встроенной справки Python.

16.2 Контрольные вопросы

1 Для чего используются функции пользователя?
2 Какой синтаксис у функций, объявляемых пользователем в языке Python?
3 Для чего используются lambda-функции?
4 Какой синтаксис у lambda-функций?

16.3 Задания для самостоятельного выполнения

1 С клавиатуры вводятся три числа – длины сторон треугольника. Напишите программу на языке программирования Python, в которой реализованы функции is_triangle – возвращает True, если треугольник существует, square – возвращает площадь треугольника, median – возвращает длину медианы, height – возвращает длину высоты, perimeter – возвращает периметр треугольника, angle – возвращает угол между двумя сторонами (в градусах).

2 С клавиатуры вводится число – количество углов многоугольника – и затем пары чисел – последовательно координаты вершин многоугольника. Напишите программу на языке программирования Python, которая определяет, является ли многоугольник выпуклым и вычисляет его площадь. Для определения выпуклости реализовать функцию to_halfplane, которая определяет положение точки относительно прямой, и функцию triangle_square, которая возвращает площадь треугольника, заданного координатами вершин.

3 С клавиатуры вводятся две пары чисел – координаты двух точек, которые определяют некоторую прямую, затем еще две пары чисел – координаты еще двух точек, которые определяют другую прямую. Напишите программу на языке программирования Python, которая реализует следующие функции: is_intersect – возвращает True, если прямые пересекаются; intersect – возвращает кортеж из двух чисел – координаты точки пересечения; intersect_angle – возвращает угол в градусах, под которым пересекаются прямые.

4 С клавиатуры вводятся три тройки чисел – координаты трех векторов в пространстве. Напишите программу на языке программирования Python, которая реализует следующие функции: is_right – возвращает True, если тройка векторов правая; is_complanar – возвращает True, если тройка векторов компланарна; pyr_volume – возвращает объем пирамиды, построенной на данных векторах.

5 С клавиатуры вводятся два числа – размерность числовой матрицы, затем вводится сама матрица. Напишите программу на языке программирования Python, которая реализует функцию, выполняющую транспонирование введенной матрицы.

Лабораторная работа № 17. Работа с текстовыми файлами

Цель работы: научить студентов использованию стандартных средств работы с файлами.

17.1 Теоретическая часть

Файл – это набор данных, сохраненных в виде последовательности битов на некотором носителе, имеющих название (имя файла). Различают два типа файлов – текстовые и бинарные. Текстовые файлы (человекочитаемые) содержат последовательности символов с некоторыми закодированными (или нет) данными. Бинарные файлы содержат данные в закодированной форме в виде битовых последовательностей (содержат только 0 или 1).

Любые файлы обрабатываются в соответствии с последовательностью:

– открытие файла;

– выполнение операций (чтение, запись);

– закрытие файла.

Для открытия файлов применяется метод open:

`open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closed=True, opener=None).`

Параметры метода:

file – идентификатор файла с указанием пути к нему в соответствии с правилами операционной системы;

mode – режим доступа, текстовый параметр, возможны следующие варианты, по умолчанию ‘r’:

– ‘r’ – открытие файла только для чтения (установлен по умолчанию);

– ‘w’ – открытие файла для записи, удаляет все данные из файла;

– ‘x’ – создание нового файла для записи;

– ‘a’ – открытие файла для записи и добавления данных в конец файла;

– ‘b’ – бинарный режим, запись битовой информации;

– ‘t’ – текстовый режим;

–‘+’ – открытие файла на диске для изменения (чтения и записи);

buffering – режим буферизации, целое число, соответствующее размеру буфера при чтении из файла, значение –1 подразумевает автоматический подбор размера буфера;

encoding – кодировка (имя кодировки) для декодирования файла;

errors – параметр, управляющий обработкой ошибок кодировки;

newline – определяет символ завершения строки (только для текстовых файлов), может принимать значения ‘\n’, ‘\r’, ‘\r\n’;

closed – если установлено значение False, после закрытия файл остается открытым;

opener – параметр, позволяющий указать альтернативный способ открытия файла.

Метод open возвращает объект класса io.TextIOWrapper. Для работы с файлом используются методы этого класса.

Для записи в файл используется метод write(text), который выводит в открытый файл текст text.

Для чтения из файла используется метод read(size=–1), который возвращает количество символов size из файла или все содержимое до конца файла. При этом курсов в файле смещается на size единиц в сторону конца файла.

Для того чтобы установить курсор в требуемую позицию, используется метод seek(pos).

В конце выполнения операций с файлом обязательно закрыть файл с помощью метода close(), однако можно использовать конструкцию with, которая закрывает файл автоматически:

```
with file=open("filename.ext", "w"):
    text=file.read()
```

17.2 Контрольные вопросы

1 Что такое файл?

2 Какой тип возвращает метод open?

3 Укажите синтаксис команды open, которая открывает файл для дополнения данными.

4 Почему предпочтительно использовать конструкцию with для работы с файлами?

17.3 Задания для самостоятельного выполнения

1 Напишите программу на языке программирования Python, которая позволяет сохранить в файл и прочитать из файла структурированные данные в виде словаря.

2 С клавиатуры вводятся n списков чисел произвольной длины (n вводится с клавиатуры). Напишите программу на языке программирования Python, которая сохраняет эти списки в файл и производит их чтение и вывод на экран из файла.

3 Имеется текстовый файл, содержащий последовательность символов. Напишите программу на языке программирования Python, которая удаляет из файла символы между 20 и 30 позициями.

4 Дан символьный файл, содержащий хотя бы один символ пробела. Напишите программу на языке программирования Python, которая удаляет все элементы, расположенные после первого пробела, включая и этот пробел.

5 Дан символьный файл, содержащий хотя бы один символ пробела. Напишите программу на языке программирования Python, которая удаляет все

элементы, расположенные после последнего пробела, включая и этот пробел.

6 Дан символьный файл, содержащий хотя бы один символ пробела. Напишите программу на языке программирования Python, которая удаляет все элементы, расположенные перед первым пробелом, включая и этот пробел.

7 Дан символьный файл, содержащий хотя бы один символ пробела. Напишите программу на языке программирования Python, которая удаляет все элементы, расположенные перед последним пробелом, включая и этот пробел.

8 Дан символьный файл. Напишите программу на языке программирования Python, которая упорядочивает элементы файла по возрастанию кодов.

9 Дано целое число K > 0 и текстовый файл. Напишите программу на языке программирования Python, которая создает два новых файла: строковый, содержащий первые K символов каждой строки исходного файла, и символьный, содержащий K-й символ каждой строки. Если длина строки меньше K, то в строковый файл записывается вся строка, а в символьный файл записывается пробел.

10 Дан строковый файл. Напишите программу на языке программирования Python, которая создает новый строковый файл, в котором строки исходного файла расположены в лексикографическом порядке.

11 Дан строковый файл, содержащий даты в формате «ДД/ММ/ГГГГ». Напишите программу на языке программирования Python, которая выводит новый файл на основе исходного с датами в формате «ГГ-ММ-ДД».

12 Дан строковый файл, содержащий даты в формате «ДД/ММ/ГГГГ». Напишите программу на языке программирования Python, которая создает два файла целых чисел, первый из которых содержит значения дней, а второй – значения месяцев для дат из исходного файла в том же порядке.

13 Дан строковый файл, содержащий даты в формате «ДД/ММ/ГГГГ». Напишите программу на языке программирования Python, которая в новый файл выводит даты в порядке возрастания, принадлежащие к некоторому времени года (вводится с клавиатуры).

14 Дан строковый файл, содержащий даты в формате «ДД/ММ/ГГГГ». Напишите программу на языке программирования Python, которая отыскивает и выводит на экран самую раннюю весеннюю дату. Если такой даты нет, на экран выводится соответствующее сообщение.

15 Дан строковый файл, содержащий даты в формате «ДД/ММ/ГГГГ». Напишите программу на языке программирования Python, которая сортирует даты в обратном порядке.

16 В текстовом файле содержится прямоугольная матрица действительных чисел, с клавиатуры вводятся два натуральных числа i и j. Напишите программу на языке программирования Python, которая выводит на экран элемент матрицы, расположенный на пересечении i-й строки и j-го столбца. Если такой элемент отсутствует, выведите 0.

17 В текстовом файле содержится прямоугольная матрица действительных чисел. Напишите программу на языке программирования Python, которая в новый файл выводит транспонированную матрицу.

18 Даны два файла вещественных чисел SA и SB, содержащие элементы прямоугольных матриц A и B (по строкам). Напишите программу на языке программирования Python, которая в новый файл SC выводит произведение матриц A*B. Если перемножение невозможно выполнить, оставить файл SC пустым.

19 Дан файл вещественных чисел, содержащий элементы верхнетреугольной матрицы (по строкам). Напишите программу на языке программирования Python, которая в новый файл выводит элементы ненулевой части данной матрицы (по строкам).

Лабораторная работа № 18. Работа с бинарными файлами

Цель работы: научить студентов методам сериализации, работы с методами библиотек pickle и json.

18.1 Теоретическая часть

Для работы с бинарными файлами используется метод open, с обязательным указанием параметра mode='b'. В стандартной поставке Python присутствуют модули pickle, csv и json, которые позволяют сохранять в бинарных файлах и восстанавливать из них любые сложные структуры данных, допускающих сериализацию.

CSV – comma separated values – текстовый формат, предназначенный для представления табличных данных. Каждая строка файла – это одна строка таблицы, где значения отдельных колонок разделяются символом (delimiter), запятой и заключаются в кавычки.

JSON – JavaScript object notation, текстовый формат обмена данными, основанный на JavaScript. Одно из преимуществ JSON – формат легко читается людьми, для его хранения в Python имеются соответствующие структуры данных.

С методами этих модулей можно ознакомиться с помощью команды help().

При использовании модуля pickle следует помнить, что недопустима десериализация данных из непроверенных источников, поскольку это может привести к порче данных или необратимым последствиям.

18.2 Контрольные вопросы

1 Что такое сериализация данных?
2 В каком виде сохраняются данные в бинарном файле?
3 Для чего служит модуль pickle?
4 Для чего служит модель csv?
5 Для чего служит модуль json?

18.3 Задания для самостоятельного выполнения

Дан телефонный справочник в формате JSON:

```
[
   {
      “имя”:”…”,
      “телефоны”:[
         {
            “описание”:”…”,
            “номер”:”…”
         },
         {
            “описание”:”…”,
            “номер”:”…”
         },
      ]
   }
]
```

Напишите программу на языке программирования Python, которая загружает справочник из бинарного файла, выполняет поиск контактов по номеру телефона, позволяет выполнять поиск контактов по имени, добавлять контакт, удалять контакты по имени, удалять номер телефона из контактов, сохранять справочник в бинарный файл.

Лабораторная работа № 19. Основы объектно-ориентированного программирования

Цель работы: ознакомить обучающихся с основами объектно-ориентированного программирования и его реализации в Python.

19.1 Теоретическая часть

Объектно-ориентированное программирование (ООП) – это парадигма программирования, в которой различные компоненты компьютерной программы моделируются по образцу реальных объектов. Объект – это все, что имеет некоторые характеристики и может выполнять определенную функцию.

Объектно-ориентированное программирование не является концепцией, зависящей от языка. Это общая концепция программирования, которую поддерживают большинство современных языков программирования.

Программа, написанная в виде объектов и классов, может быть повторно использована в других проектах. Модульный подход, используемый в объектно-ориентированном программировании, приводит к высокому уровню обслуживания кода. В объектно-ориентированном программировании каждый

класс решает определенную задачу. Если ошибка возникает в одной части кода, ее можно исправить локально, не затрагивая другие части кода.

Инкапсуляция данных добавляет дополнительный уровень безопасности программе, разработанной с использованием ООП.

Основной структурной единицей объектно-ориентированного кода является класс. Класс в ООП служит чертежом объекта. Каждый объект является экземпляром класса. Для объявления класса в Python используется следующая инструкция:

```
class name(object):
    поля и методы класса name
```

Процесс создания объекта класса называется instantiation. В Python для создания объекта класса необходимо написать имя класса, за которыми следуют скобки, например, инструкция

```
obj=name()
```

создаст объект класса name. Для вызова методов или использования атрибутов класса необходимо написать имя объекта и через точку название атрибута или метода.

В Python каждый объект имеет некоторые атрибуты и методы по умолчанию в дополнение к пользовательским атрибутам. Для просмотра всех атрибутов и методов объекта можно использовать встроенную функцию dir().

Атрибуты можно разделить на атрибуты класса и атрибуты экземпляра. Атрибуты класса являются для всех объектов класса, в то время как атрибуты экземпляра являются исключительным свойством экземпляра. Атрибуты экземпляра объявляются внутри любого метода, в то время как атрибуты класса объявляются вне любого метода.

Методы в ООП используются для реализации функционала объекта. Методы объявляются как функции внутри описания класса. Некоторые методы могут быть вызваны непосредственно с помощью имени класса, такие методы называются статическими.

Чтобы объявить статический метод, можно указать декоратор @staticmethod перед именем метода:

```
class Name:
    @staticmethod
    def get_something():
        print("something")
```

В данном случае метод get_something может быть вызван не как метод объекта, а прямым указанием имени класса Name.get_something().

Методы класса в Python позволяют возвращать несколько значений.

Для вывода объектов класса на печать необходимо переопределить стандартный метод классов __str__, предопределенный языком программирования. С помощью этого метода можно создавать пользовательские представления для вывода объекта на печать.

Конструктор – это специальный метод, который вызывается по умолчанию всякий раз, когда Вы создаете объект класса. Чтобы создать конструктор, необходимо создать метод с ключевым словом __init__. При создании объекта

класса будут выполнены действия, описанные в конструкторе. Конструктор может использоваться как обычный метод, с ним можно обмениваться данными. Обычно конструктор используется, если необходимо инициализировать значения атрибутов при создании экземпляра класса.

Атрибуты класса по сути являются переменными. В зависимости от области действия переменные делятся на локальные и глобальные.

Локальная переменная в классе – это переменная, доступ к которой возможен только внутри блока кода (описания метода), в котором она определена.

Глобальная переменная определяется вне любого блока кода. Глобальная переменная может быть доступна в любом месте класса.

Модификаторы доступа в Python используются для изменения области действия переменных по умолчанию. В Python существуют три типа модификаторов (декораторов) доступа: public, private и protected.

Переменные с декораторами открытого доступа могут быть доступны в любом месте внутри или вне класса, приватные переменные могут быть доступны только внутри класса, защищенные переменные могут быть доступны в том же пакете. Чтобы создать закрытую переменную, необходимо префиксировать двойные подчеркивания с именем переменной. Чтобы создать защищенную переменную, необходимо добавить префикс одного подчеркивания с именем переменной. Имена открытых переменных не префиксируются.

Наследование в ООП состоит в том, что класс может наследовать характеристики другого класса. Класс, который наследует другой класс, называется потомком, а класс, наследуемый другим классом, – родительским или базовым классом. В Python родительский класс может иметь несколько классов-потомков и, наоборот, класс-потомок может иметь несколько родительских классов.

Если необходимо изменить к инициализатору класса-родителя, необходимо использовать метод super(). Вызывать родительский метод следует в первую очередь.

Полиморфизм в ООП относится к способности объекта вести себя несколькими способами. В программировании полиморфизм реализуется через перегрузку методов и переопределение методов.

Перегрузка метода относится к свойству метода вести себя по-разному в зависимости от количества или типа параметров. Например, метод method класса Name

```
class Name:
    def method(self, a, b=None):
        if b is None:
            print(a)
        else:
            print(a+b)
```

будет вести себя по-разному в случае, если вызывать его с одним или двумя параметрами. В этом примере ключевое слово self передает ссылку на текущий объект, для которого этот метод был вызван. Слово self не является зарезервированным, в качестве первого параметра метода класса может быть любое другое слово.

Переопределение метода означает наличие метода с тем же именем в классе-потомке, что и в родительском. Определение метода отличается в родительском и дочернем классах, но имя остается тем же.

Инкапсуляция в ООП подразумевает, что сложность реализации программного компонента должна быть спрятана за его интерфейсом.

19.2 Контрольные вопросы

1 Что такое класс в ООП?
2 Что такое экземпляр класса?
3 Что такое метод класса?
4 В чем отличие атрибутов класса и экземпляра?
5 В чем проявляется полиморфизм?
6 Какие уровни доступа к данным существуют в Python и как они используются?

19.3 Задания для самостоятельного выполнения

1 Есть **Человек**, характеристиками которого являются: имя, возраст, наличие денег, наличие собственного жилья. Человек может предоставить информацию о себе, заработать деньги, купить дом. Также есть **Дом**, свойства которого – площадь и стоимость. Для дома можно применить скидку на покупку. Также есть **Небольшой Типовой Дом**, обязательной площадью 40 кв. м. Напишите программу на языке программирования Python, которая реализует классы Human, House и SmallHouse.

Класс Human имеет два статических поля default_name и default_age. Создайте метод __init__(), который принимает два параметра name и age. Для этих параметров задайте значения по умолчанию, используя свойства default_name и default_age. В методе __init__() определите четыре свойства: публичные name и age, приватные money и house. Реализуйте справочный метод default_info(), который будет выводить статические поля default_name и default_age. Реализуйте приватный метод make_deal(), который будет отвечать за техническую реализацию покупки дома: уменьшать количество денег на счету и присваивать ссылку на только что купленный дом. В качестве аргументов данный метод принимает объект дома и его цену. Реализуйте метод earn_money(), увеличивающий значение свойства money. Реализуйте метод buy_house(), который будет проверять, что у человека достаточно денег для покупки, и совершать сделку. Если денег недостаточно – нужно вывести предупреждение. Параметры метода: ссылка на дом и размер скидки.

Класс House. Создайте метод __init__() и определите внутри него два динамических свойства _area и _price. Свои начальные значения они получают из параметров метода __init__(). Создайте метод final_price(), который принимает в качестве параметра размер скидки и возвращает цену с учетом данной скидки.

Класс SmallHouse наследует функционал от класса House. Внутри класса

SmallHouse переопределите метод __init__() так, чтобы он создавал объект с площадью 40 м^2.

Проверьте выполнение, используя тесты:

– вызовите справочный метод default_info() класса Human;

– создайте объект класса Human;

– выведите справочную информацию о созданном объекте вызовом метода info();

– создайте объект класса SmallHouse;

– попробуйте купить созданный дом, убедитесь в получении сообщения;

– поправьте финансовое положение объекта – вызовите метод earn_money();

– снова попробуйте купить дом;

– посмотрите, как изменилось состояние объекта Human.

2 Есть **Алфавит**, характеристиками которого являются Язык и Список букв. Для **Алфавита** можно напечатать все буквы алфавита, посчитать количество букв. Также есть **Английский алфавит**, который обладает следующими свойствами: Язык, Список букв, Количество букв. Для **Английского алфавита** можно посчитать количество букв, определить, относится ли буква к английскому алфавиту, получить пример текста на английском языке.

Создайте класс **Alphabet.** Создайте метод **__init__()**, внутри которого будут определены два динамических свойства **lang** – язык и **letters** – список букв. Начальные значения свойств берутся из входных параметров метода.

Создайте метод **print()**, который выведет в консоль буквы алфавита.

Создайте метод **letters_num()**, который вернет количество букв в алфавите.

Создайте класс **EngAlphabet** путем наследования от класса **Alphabet.** Создайте метод **__init__()**, внутри которого будет вызываться родительский метод **__init__()**. В качестве параметров ему будут передаваться обозначение языка (например, **'En'**) и строка, состоящая из всех букв алфавита (можно воспользоваться свойством **ascii_uppercase** из модуля **string**).

Добавьте приватное статическое свойство **__letters_num**, которое будет хранить количество букв в алфавите.

Создайте метод **is_en_letter()**, который будет принимать букву в качестве параметра и определять, относится ли эта буква к английскому алфавиту.

Переопределите метод **letters_num()** – пусть в текущем классе классе он будет возвращать значение свойства **__letters_num**.

Создайте статический метод **example()**, который будет возвращать пример текста на английском языке.

Тесты:

– создайте объект класса **EngAlphabet**;

– напечатайте буквы алфавита для этого объекта;

– выведите количество букв в алфавите;

– проверьте, относится ли буква **F** к английскому алфавиту;

– проверьте, относится ли буква **Щ** к английскому алфавиту;

– выведите пример текста на английском языке.

3 Есть **Помидор** со следующими характеристиками: Индекс, Стадия зрелости (стадии: Отсутствует, Цветение, Зеленый, Красный). **Помидор** может

расти (переходить на следующую стадию созревания), предоставлять информацию о своей зрелости.

Есть **Куст с помидорами**, который имеет свойство Содержит список томатов, которые на нем растут. Может расти вместе с томатами, предоставлять информацию о зрелости всех томатов, предоставлять урожай.

И также есть **Садовник**, который имеет Имя, Растение, за которым он ухаживает. Может ухаживать за растением, собирать с него урожай.

Создайте класс **Tomato.** Создайте статическое свойство **states**, которое будет содержать все стадии созревания помидора. Создайте метод **__init__()**, внутри которого будут определены два динамических protected свойства: **_index –** передается параметром и **_state** – принимает первое значение из словаря **states.** Создайте метод **grow()**, который будет переводить томат на следующую стадию созревания. Создайте метод **is_ripe()**, который будет проверять, что томат созрел (достиг последней стадии созревания).

Создайте класс **TomatoBush.** Определите метод **__init__()**, который будет принимать в качестве параметра количество томатов и на его основе будет создавать список объектов класса **Tomato**. Данный список будет храниться внутри динамического свойства **tomatoes**. Создайте метод **grow_all()**, который будет переводить все объекты из списка томатов на следующий этап созревания. Создайте метод **all_are_ripe()**, который будет возвращать **True**, если все томаты из списка стали спелыми. Создайте метод **give_away_all()**, который будет чистить список томатов после сбора урожая.

Создайте класс **Gardener.** Создайте метод **__init__()**, внутри которого будут определены два динамических свойства **name** – передается параметром, является публичным и **_plant** – принимает объект класса **Tomato**, является **protected.** Создайте метод **work()**, который заставляет садовника работать, что позволяет растению становиться более зрелым. Создайте метод **harvest()**, который проверяет, все ли плоды созрели. Если все – садовник собирает урожай, если нет – метод печатает предупреждение. Создайте статический метод **knowledge_base()**, который выведет в консоль справку по садоводству.

Тесты:

– вызовите справку по садоводству;

– создайте объекты классов **TomatoBush** и **Gardener**;

– используя объект класса **Gardener**, поухаживайте за кустом с помидорами;

– попробуйте собрать урожай;

– если томаты еще не дозрели, продолжайте ухаживать за ними;

– соберите урожай.

Лабораторная работа № 20. Работа с графикой

Цель работы: изучить особенности построения графических интерфейсов и графики с помощью стандартных библиотек Python.

20.1 Теоретическая часть

Для реализации графических интерфейсов и графических примитивов в стандартной поставке Python присутствует библиотека tkinter, которая является мощным инструментом, позволяющим реализовать графический интерфейс пользователя или изобразить графический материал. Например, редактор кода IDLE, входящий в стандартную поставку Python, разработан с применением tkinter. Среди преимуществ библиотеки кроссплатформенность – модуль использует нативные графические элементы операционных систем, поэтому созданное с помощью tkinter визуальное приложение будет выглядеть так, как будто разработано специально для использования именно в этой операционной системе.

Концепция работы с модулем tkinter достаточно проста: главным элементом интерфейса является окно. Окнами называют контейнеры, в которых находятся все элементы графического интерфейса. Данные элементы, к числу которых относятся текстовые боксы, ярлыки и кнопки, называются **виджетами**. Виджеты помещаются внутри окон.

Tk является базовым классом любого Tkinter-приложения. При создании объекта этого класса запускается интерпретатор tcl/tk и создается базовое окно приложения. Tkinter является событийно-ориентированной библиотекой. В приложениях такого типа имеется главный цикл обработки событий. В Tkinter такой цикл запускается методом mainloop. Для явного выхода из интерпретатора и завершения цикла обработки событий используется метод quit.

В приложении можно использовать несколько интерпретаторов tcl/tk. Так как после вызова метода mainloop дальнейшие команды python исполняться не будут до выхода из цикла обработки событий, необходимо метод mainloop всех интерпретаторов, кроме последнего, осуществлять в фоновом режиме.

При использовании двух и более интерпретаторов необходимо следить, чтобы объекты, созданные в одном интерпретаторе, обрабатывались только в нём. Например, изображение, созданное в первом интерпретаторе, может быть использовано много раз в этом же интерпретаторе, но не может быть использовано в других интерпретаторах. Необходимость в запуске нескольких интерпретаторов в одном приложении возникает крайне редко. Для создания дополнительного окна приложения в большинстве случаев достаточно виджета Toplevel.

Все виджеты в Tkinter обладают некоторыми общими свойствами. Виджеты создаются вызовом конструктора соответствующего класса. Первый аргумент (как правило, неименованный, но можно использовать имя *master*) – это родительский виджет, в который будет упакован (помещён) генерируемый виджет. Родительский виджет можно не указывать, в таком случае будет

использовано главное окно приложения. Далее следуют именованные аргументы, конфигурирующие виджет. Это может быть используемый шрифт (font=...), цвет виджета (bg=...), команда, выполняющаяся при активации виджета (command=...) и т. д.

Виджеты могут быть сконфигурированы во время создания, но иногда необходимо изменить конфигурацию виджета во время исполнения программы. Для этого используется метод *configure* (или его синоним *config*). Также можно использовать квадратные скобки (widget['option'] = new_value).

Метод cget является обратным к методу configure. Он предназначен для получения информации о конфигурации виджета. Здесь, как и в случае с configure, можно использовать квадратные скобки (value = widget['option']).

Метод destroy выполняет уничтожение виджета и всех его потомков. Следует отметить, что если необходимо только на время спрятать какой-либо виджет, то лучше использовать упаковщик grid и метод grid_remove.

Методы семейства grab_ предназначены для управления потоком события. Виджет, захвативший поток, будет получать все события окна или приложения.

grab_set – передает поток данному виджету.

grab_set_global – передает **глобальный** поток данному виджету. В этом случае все события на дисплее будут передаваться этому виджету. Следует использовать очень осторожно, т. к. остальные виджеты всех приложений не будут получать события.

grab_release – освобождает поток.

grab_status – возвращает текущий статус потока событий для виджета. Возможные значения: None, "local" или "global".

grab_current – возвращает виджет, который получает поток.

Методы семейства focus_ используются для управления фокусом ввода с клавиатуры. Виджет, имеющий фокус, получает все события с клавиатуры.

focus (синоним focus_set) – передать фокус виджету.

focus_force – передать фокус, даже если приложение не имеет фокуса. Используйте осторожно, поскольку это может раздражать пользователей.

focus_get – возвращает виджет, на который направлен фокус, либо None, если такой отсутствует.

focus_displayof – возвращает виджет, на который направлен фокус на том дисплее, на котором размещён виджет, либо None, если такой отсутствует.

focus_lastfor – возвращает виджет, на который будет направлен фокус, когда окно с этим виджетом получит фокус.

tk_focusNext – возвращает виджет, который получит фокус следующим (обычно смена фокуса происходит при нажатии клавиши Tab). Порядок следования определяется последовательностью упаковки виджетов.

tk_focusPrev – то же, что и focusNext, но в обратном порядке.

tk_focusFollowsMouse – устанавливает, что виджет будет получать фокус при наведении на него мышью. Вернуть нормальное поведение достаточно сложно.

К методам, не являющимся виджет-специфичными, относятся таймеры. С помощью этих методов можно отложить выполнение какого-нибудь кода на определённое время.

after – принимает два аргумента: время в миллисекундах и функцию, которую надо выполнить через указанное время. Возвращает идентификатор, который может быть использован в after_cancel.

after_idle – принимает один аргумент – функцию. Эта функция будет выполнена после завершения всех отложенных операций (после того как будут обработаны все события). Возвращает идентификатор, который может быть использован в after_cancel.

after_cancel – принимает один аргумент – идентификатор задачи, полученный предыдущими функциями, – и отменяет это задание.

Основные виджеты tkinter:

– TopLevel – окно верхнего уровня, используется для создания многооконных программ и диалоговых окон;

– Button – кнопка;

– Label – виджет для отображения надписи без возможности редактирования пользователем;

– Entry – виджет, позволяющий пользователю ввести одну строку текста;

– Text – виджет, позволяющий ввести любое количество текста;

– ListBox – виджет, представляющий собой список, из элементов которого пользователь может выбрать один или несколько пунктов;

– Frame – рамка, предназначена для организации виджетов внутри окна;

– Checkbutton – виджет, позволяющий отметить «галочкой» определенный пункт в окне;

– Radiobutton – позволяет пользователю выбрать только один из предложенных пунктов;

– Scale – шкала – виджет, позволяющий выбрать какое-либо значение из диапазона;

– Scrollbar – виджет для «прокрутки» другого виджета.

Более подробно с методами виджетов можно ознакомиться, используя справочную систему Python.

Упаковщики или менеджеры расположения pack, place, grid представляют пользователю специальный механизм, который размещает виджеты на окне. Недопустимо использовать в одном приложении различные упаковщики.

Упаковщик pack() требует в свойстве side указания, к какой стороне родительского виджета он должен примыкать. Синоним pack_configure().

Упаковщик grid() представляет собой таблицу с ячейками, в которые помещаются виджеты, имеет параметры, определяющие таблицу виджетов и место расположения конкретного виджета.

Упаковщик place() позволяет разместить виджет в фиксированном месте с фиксированным размером. Позволяет указывать координаты размещения в относительных единицах. Требует указывать координаты каждого размещаемого виджета.

Для работы с двумерной графикой в tkinter базовым понятием является

холст – класс Canvas(), позволяющий располагать геометрические фигуры, изображения или другие виджеты. Обязательными параметрами являются размеры холста.

Для корректного размещения виджетов или графических объектов следует понимать, что нулевая точка с координатами (0, 0) будет располагаться в верхнем левом углу холста.

Графические примитивы tkinter:

– create_line – создает линию на холсте;

– create_rectangle – создает прямоугольник;

– create_polygon – создает многоугольник, заданный парами координат его вершин;

– create_oval – создает эллипс, в качестве параметров принимает координаты вершин гипотетического прямоугольника, в который можно вписать эллипс;

– create_arc – служит для создания сектора, сегмента или дуги (определяется параметром style);

– create_text – создает текстовую запись.

Каждый объект при создании возвращает идентификатор, который можно связать с переменными и использовать для обращения к конкретному объекту.

Объекты можно перемещать с помощью модификатора move, в качестве параметров метод получает объект и относительные смещения по осям.

Метод tag_bind позволяет привязать событие (например, щелчок кнопкой мыши) к определенному объекту.

Подробная справочная информация по классам и методам модуля tkinter доступна на сайте https://ru.wikibooks.org/wiki/GUI_Help/Tkinter_book.

Для построения анимации следует использовать метод Canvas.update(), позволяющий перерисовывать виджеты на каждом шаге основного цикла.

20.2 Контрольные вопросы

1 Для чего служит класс Tk() модуля tkinter?

2 Можно ли с помощью методов класса tkinter реализовать графический интерфейс пользователя?

3 Какой класс модуля tkinter необходимо использовать для вывода двумерной графики?

20.3 Задания для самостоятельного выполнения

В данной программе создается анимация круга, который движется от левой границы холста до правой:

```
from tkinter import *
root = Tk()
c = Canvas(root, width=300, height=200, bg="white")
c.pack()
ball = c.create_oval(0, 100, 40, 140, fill='green')
```

```
def motion():
    c.move(ball, 1, 0)
    if c.coords(ball)[2] < 300:
        root.after(20, motion)
motion()
root.mainloop()
```

1 Изучите программу, приведенную в качестве примера и запрограммируйте постепенное движение фигуры в ту точку холста, где пользователь кликает левой кнопкой мыши. Координаты события хранятся в его атрибутах x и y (event.x, event.y).

2 Напишите программу на языке программирования Python, которая реализует постепенное движение двух фигур (круги красного и зеленого цвета) в те точки холста, где пользователь кликает левой или правой кнопками мыши.

3 На холсте размером 800*600 пикселей расположен зеленый круг радиусом 50 пикселей. Напишите программу на языке программирования Python, которая реализует движение круга по диагонали и отскок от границ холста по принципу «угол падения равен углу отражения».

Список литературы

1 **Гуриков, С. Р.** Основы алгоритмизации и программирования на Python: учебное пособие / С. Р. Гуриков. – Москва : ФОРУМ; ИНФРА-М, 2020. – 343 с.

2 **Гэддис, Т.** Начинаем программировать на Python: пер. с англ. / Т. Гэддис. – 4-е изд. – Санкт-Петербург : БХВ-Петербург, 2019. – 768 с.

3 **Жуков, Р. А.** Язык программирования Python: практикум: учебное пособие / Р. А. Жуков. – Москва : ИНФРА-М, 2020. – 216 с.

4 **Златопольский, Д. М.** Основы программирования на языке Python / Д. М. Златопольский. – Москва : ДМК Пресс, 2017. – 284 с.

5 **Лутц, М.** Изучаем Python: в 2 т. / М. Лутц. – Санкт-Петербург: Диалектика, 2019.

6 **Прохоренок, Н. А.** Python 3. Самое необходимое / Н. А. Прохоренок, В. А. Дронов. – 2-е изд., перераб. и доп. – Санкт-Петербург: БХВ-Петербург, 2018. – 608 с.